UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Mathematical programming modelling of the response time variability problem

Albert Corominas, Wieslaw Kubiak and Rafael Pastor

EOLI: Enginyeria d'Organització i Logística Industrial

IOC-DT-P-2006-17
Juny 2006

Institut d'Organització i Control
de Sistemes Industrials

Mathematical Programming Modelling of the Response Time Variability Problem[†]

Albert Corominas[1], Wieslaw Kubiak[2] and Rafael Pastor[1,*]
[1]Institute of Industrial and Control Engineering (IOC), Catalonia University of Technology, Barcelona, Spain
[2]Faculty of Business Administration, Memorial University of Newfoundland, St. John's, Canada

Abstract

The Response Time Variability Problem (RTVP) is a scheduling problem that has recently been defined in the literature. The RTVP has a broad range of real-life applications. For example, in the automobile industry it can be used to sequence the models to be produced on a mixed-model assembly line. A previous study developed a position exchange heuristic to apply to certain greedy initial sequences for the RTVP. Some mathematical programming models (MILPs) have also been tested to solve it to optimality, but the practical limit for obtaining optimal solutions is 25 units to be scheduled. This paper aims to improve the best mathematical programming model developed thus far in order to solve larger instances up to 40 units to optimality. The contribution of this paper is threefold: i) larger instances can be solved to optimality; ii) the new optimal solutions of the RTVP can be used to compare the results of heuristic procedures; and iii) the importance of modelling is demonstrated, as well as the huge impact that reformulation, redundant constraints and the elimination of symmetries have on the efficiency of MILPs.

Keywords: response time variability, fair sequences, optimisation, scheduling

1. Introduction

Corominas *et al*. (2004) have recently introduced the Response Time Variability Problem (RTVP) to model a broad range of real-life situations that occur whenever events, jobs, clients or products need to be sequenced so as to minimise the variability of the time they wait for their next turn in obtaining the resources they need to advance. For example, it can be used in the automobile industry to sequence the models to be produced on a mixed-model assembly line (Monden, 1983). The solution to the RTVP problem also solves the periodic machine maintenance problem (Anily *et al*., 1998 and Bar-Noy *et al*. 2002), it can be used to optimize the stride scheduling technique (Waldspurger and Weihl, 1995) as well as other distance-constrained problems (e.g., see Han *et al*. 1996 and Dong *et al*. 1998).

The RTVP is quite simple to formulate but rather difficult to solve to optimality. Let n be the number of products/jobs/messages (in this paper we will only use the term "product"). Let d_i be the number of units of product i $(i=1,...,n)$ to be scheduled in a sequence $s=s_1s_2...s_D$ of length D $\left(D=\sum_{i=1}^{n} d_i\right)$, i.e., with D positions. For all types of

[†] Supported by the Spanish MCyT project DPI2004-03472, co-financed by FEDER and by the Natural Sciences and Engineering Research Council of Canada Research Grant OGP0105675.
[*] Corresponding author: Rafael Pastor, Institut d'Organització i Control de Sistemes Industrials (IOC), Av. Diagonal 647 (edif. ETSEIB), planta 11, 08028 Barcelona, Spain; Tel. + 34 93 401 17 01; Fax. + 34 93 401 66 05; e-mail: rafael.pastor@upc.edu

product with $d_i \geq 2$, let t_k^i be the distance between the position of units $k+1$ and k of product i. Let us assume that s_1 immediately follows s_D (i.e., it is a circular sequence). Therefore, $t_{d_i}^i$ is the distance between the first unit of product i in a cycle and the last unit of the same product in the preceding cycle. Let $\overline{t}_i = D/d_i$ be the average distance between two consecutive units of product i. The objective is to minimise $RTV = \sum_{i=1}^{n}\sum_{k=1}^{d_i}\left(t_k^i - \overline{t}_i\right)^2$, which is a weighted variance with weights being equal to demands: $RTV = \sum_{i=1}^{n} d_i \cdot Var_i$, where $Var_i = \frac{1}{d_i}\cdot\sum_{k=1}^{d_i}\left(t_k^i - \overline{t}_i\right)^2$.

Corominas *et al*. (2004) study the RTVP computational complexity and prove that the RTVP is NP-complete. They propose an optimization algorithm for a two-product case. In order to solve the RTVP to optimality they consider a special case of the quadratic assignment problem recast as a quadratic integer programming (QIP) problem. The QIP is linearised using three different techniques, but the practical limit for getting optimal solutions with the best MILP obtained, is 25 units to be scheduled (i.e., $D = 25$). Finally, a simple position exchange heuristic is presented for application to some greedy initial sequences and a computational experiment using the heuristic is reported.

León *et al*. (2003) present a classification of the Production Rate Variation min-var problem, which also includes the RTVP. Some mathematical programming models for the resulting classification problems are also included (see León *et al*. (2003) for more details).

This paper builds on the Corominas *et al*. (2004) study. It analyses some special features of the RTVP and presents some new ideas for improving the MILP model used to solve the RTVP optimally. The objective of this paper, therefore, is to improve the best mathematical programming model presented in the literature thus far in order to solve larger instances to optimality. The contribution of this paper is threefold: i) larger instances can be solved to optimality for a problem that has numerous real-life applications; ii) the new optimal solutions of the RTVP can be used to compare the results of heuristic procedures, which are needed to solve medium-large instances; and iii) the importance of modelling is demonstrated, as well as the huge impact that reformulation, redundant constraints and the elimination of symmetries have on the effectiveness of MILPs.

The remainder of the paper is set out as follows. First, Section 2 presents the terminology, the best MILP model presented in the literature and a lower bound on the value of the objective function. Section 3 introduces the ideas for improving the MILP. Section 4 presents the results of the subsequent computational experiment. Finally, Section 5 is devoted to conclusions and possible venues for future research.

2. Terminology, initial MILP and a lower bound

This section presents the main terminology that will be used in this paper. It explains the MILP for the RTVP presented in Corominas *et al*. (2004), referred to as MILP-0 in this

paper, and describes a lower bound on the value of the objective function RTV given by Corominas *et al*., (2004).

Data:

n	Number of products $(i=1,...,n)$
D	Number of positions in the sequence
d_i	Number of units of product i $(i=1,...,n)$ to be scheduled; it is assumed that $\sum_{i=1}^{n} d_i = D$
$\overline{t}_i$	Average distance between two consecutive units of product i: $\overline{t}_i = D/d_i$ $(i=1,...,n)$
$G1$	Set of products with $d_i \geq 2$: $G1 = \{i \mid d_i \geq 2\}$
UB_i	Upper bound on the distance between two consecutive units of product i: $UB_i = D - d_i + 1$ $(\forall i \in G1)$
E_{ik}, L_{ik}	First and last position that can be occupied by unit k of product i: $E_{ik} = k$ and $L_{ik} = D - d_i + k$ $(i=1,...,n; k=1,...,d_i)$
H_{ik}	Set of positions that can be occupied by unit k of product i: $H_{ik} = \{h \mid E_{ik} \leq h \leq L_{ik}\}$ $(i=1,...,n; k=1,...,d_i)$

Variables:

$y_{ikh} \in \{0,1\}$	1 if and only if unit k of product i is placed in position h $(i=1,...,n;\ k=1,...,d_i;\ h \in H_{ik})$
$\delta_{ik}^{j} \in \{0,1\}$	1 if and only if the distance between units k and $k+1$ of product i is equal to j $(\forall i \in G1;\ k=1,...,d_i;\ j=1,...,UB_i)$

Model MILP-0:

$$[MIN]\, RTV = \sum_{\forall i \in G1, k, j} j^2 \cdot \delta_{ik}^{j} - \sum_{i \in G1} d_i \cdot \overline{t}_i^{\,2} \tag{1}$$

$$\sum_{\forall (i,k) \mid h \in H_{ik}} y_{ikh} = 1 \qquad (h=1,...,D) \tag{2}$$

$$\sum_{h \in H_{ik}} y_{ikh} = 1 \qquad (i=1,...,n; k=1,...,d_i) \tag{3}$$

$$\sum_{h \in H_{i,k+1}} h \cdot y_{i,k+1,h} - \sum_{h \in H_{ik}} h \cdot y_{ikh} = \delta_{ik}^{1} + ... + j \cdot \delta_{ik}^{j} + ... + UB_i \cdot \delta_{ik}^{UB_i} \qquad (\forall i \in G1; k=1,...,d_i - 1) \tag{4}$$

$$D - \sum_{h \in H_{id_i}} h \cdot y_{i,d_i,h} + \sum_{h \in H_{i1}} h \cdot y_{i1h} = \delta_{id_i}^{1} + ... + j \cdot \delta_{id_i}^{j} + ... + UB_i \cdot \delta_{id_i}^{UB_i} \qquad (\forall i \in G1) \tag{5}$$

$$\sum_{j=1}^{UB_i} \delta_{ik}^{j} = 1 \qquad (\forall i \in G1; k=1,...,d_i) \tag{6}$$

Objective function (1) minimises the response time variability. Constraints (2) ensure that one and only one unit is placed in each position h of the sequence. Constraint set (3) ensures that each unit k of each type of product i is assigned to one and only one position of the sequence. Constraints (4) and (5) ensure that the distance between units k and $k+1$ of product i is equal to an integer value $j \in [1, UB_i]$ ((5) refers to the distance between the first unit of product i in a cycle and the last unit of the same product in the preceding cycle). Finally, constraint set (6) ensures that the distance between units k and $k+1$ of product i is obtained with one and only one integer value.

The original non-linear objective function $RTV = \sum_{i=1}^{n} \sum_{k=1}^{d_i} \left(t_k^i - \overline{t}_i \right)^2$ is linearised straightforwardly by defining the binary variables y_{ikh} and δ_{ik}^j. The products with $d_i = 1$ $(i \notin G1)$ are not considered in the objective function or in the related constraints – (4), (5) and (6) – because their contributions to RTV values are equal to zero.

As introduced in Corominas *et al.* (2004), for product $i \in G1$ a decomposition vector of D into d_i components can be defined as follows: $\lambda = (\lambda_1, ..., \lambda_{d_i})$ of d_i positive integers that add up to D and $\lambda_1 \geq ... \geq \lambda_{d_i}$. The components of vector λ can be the distances between the d_i units of product i. Thus, the minimum value of RTV for product i, RTV_i, can be obtained when $D \bmod d_i$ and $d_i - D \bmod d_i$ components of λ are equal to $\left\lceil \frac{D}{d_i} \right\rceil$ and $\left\lfloor \frac{D}{d_i} \right\rfloor$ respectively (they are the best-fitted values to the average distance $\overline{t}_i$). For example, let $D = 19$ and $d_i = 4$ and let the decomposition vector $\lambda = (5,5,5,4)$ provide the minimum value of RTV_i. A lower bound on the value of RTV_i, TLB_i, and a lower bound on the value of RTV, TLB, can be defined as follows: $TLB_i = (D \bmod d_i) \cdot \left(\left\lceil \frac{D}{d_i} \right\rceil - \overline{t}_i \right)^2 + (d_i - D \bmod d_i) \cdot \left(\left\lfloor \frac{D}{d_i} \right\rfloor - \overline{t}_i \right)^2$ and $TLB = \sum_{\forall i \in G1} TLB_i$.

By analysing the structure of TLB_i, one can straightforwardly define an α-constrained lower bound on the value of RTV_i, αTLB_i, when the distance between two units is fixed to α (in this situation, the decomposition vector λ' of a new RTV sub-problem, with $D' = D - \alpha$ and $d_i' = d_i - 1$, must be obtained). For example, let $D = 19$, $d_i = 4$ and $\alpha = 9$: the decomposition vector $\lambda = (9,4,3,3)$ gives αTLB_i $\left(\alpha TLB_i = \left(9 - \frac{19}{4}\right)^2 + \left(4 - \frac{19}{4}\right)^2 + \left(3 - \frac{19}{4}\right)^2 + \left(3 - \frac{19}{4}\right)^2 \right)$, where $\lambda = (9, \lambda')$ and $\lambda' = (4,3,3)$ is the decomposition vector of $D' = 19 - 9 = 10$ and $d_i' = 4 - 1 = 3$.

3. Ideas to improve model MILP-0

The ideas tested to improve the performance of model MILP-0 fall into the following five categories and are presented in Sections 3.1 to 3.5 respectively: i) tightening the

definition of the data and defining new data, ii) eliminating symmetries, iii) introducing redundant constraints, iv) setting the parameters of CPLEX (ILOG CPLEX 9.0 is the optimiser used to test the MILP to optimality), and v) other strategies. Henceforth we will refer to the ideas proposed as TD_x, ES_x, RC_x, SP_x and OS_x respectively.

First, we present the additional terminology used in this section:

Data:

LB_i	Lower bound on the distance between two consecutive units of product i $(\forall i \in G1)$
TLB_i	Lower bound on the value of the objective function for product i $(\forall i \in G1)$
TLB	Lower bound on the value of RTV: $TLB = \sum_{\forall i \in G1} TLB_i$
αTLB_i	α-constrained lower bound on the value of RTV for product i $(\forall i \in G1)$
$\bar{X}$	Initial feasible solution; $\bar{X}$ is the best solution obtained by applying a position exchange heuristic to five greedy initial sequences (see Corominas *et al*. (2004) for details)
Z	RTV value of $\bar{X}$
Z_i	Upper bound on the contribution of product i to Z: $Z_i = Z - \sum_{\forall j \in G1 \mid j \neq i} TLB_j$
s, d_s	If the number of products with $d_i = 1$ is ≥ 2, they are collected in a single fictitious product s with a number of units $d_s = n - \lvert G1 \rvert$
S_{ik}	Sets of pairs of positions that can be occupied by units k and $k+1$ of product i respectively, $S_{ik} = \{(h, j) \mid (h \in H_{ik}, j \in H_{i,k+1}) \wedge ((LB_i \leq j - h) \wedge (j - h \leq UB_i))\}$ $(\forall i \in G1; k = 1, ..., d_i - 1)$
S_{id_i}	Sets of pairs of positions that can be occupied by units d_i and 1 of product i respectively, $S_{id_i} = \{(h, j) \mid (h \in H_{id_i}, j \in H_{i1}) \wedge ((LB_i \leq D + j - h) \wedge (D + j - h \leq UB_i))\}$ $(\forall i \in G1)$

Variables:

sl_{ik}	Position of unit k of product i $(\forall i \in G1; k = 1, ..., d_i)$
$x_{ikh} \in \{0,1\}$	1 if and only if unit k of product i is placed in the position h or in a previous position $(i = 1, ..., n; k = 1, ..., d_i; h = E_{ik}, ..., L_{ik} - 1)$; note that $x_{i,k,L_{ik}} = 1$ and is not defined
$y1_h \in \{0,1\}$	1 if and only if one unit of any product with $d_i = 1$ is placed in position h $(h = 1, ..., D)$

$\gamma_{ik}^{j} \in \{0,1\}$ 1 if and only if the distance between units k and $k+1$ of product i is greater than or equal to $LB_i + j$ $(\forall i \in G1; k = 1,...,d_i;\ j = 1,...,UB_i - LB_i)$

3.1. Tightening the definition of the data and defining new data

TD$_1$: LB_i is defined and used. LB_i has an obvious value, $LB_i = 1$, but it could be tightened. The smallest integer α that satisfies $(\overline{t_i} - \alpha)^2 \le Z_i$ is calculated. Then, the α-constrained lower bound on the value of the objective function for product i, αTLB_i, is calculated: if $\alpha TLB_i \le Z_i$ then $LB_i = \alpha$, otherwise $\alpha = \alpha + 1$ and so on. Let us present a brief example: Let $D = 52$, $d_i = 5$, $\overline{t_i} = 52/5 = 10.4$ and $Z_i = 33.2$. Then, $\alpha = 5$ and, when the distance between two units is fixed to $\alpha = 5$, the best decomposition vector is $\lambda = (12,12,12,11,5)$, which gives $\alpha TLB_i = 37.2 > Z_i$. Then, $\alpha = 5+1 = 6$ and the best decomposition vector is $\lambda = (12,12,11,11,6)$, which gives $\alpha TLB_i = 25.2 \le Z_i$ and $LB_i = 6$.

TD$_2$: If LB_i is known, UB_i is tightened as follows: $UB_i = D - ((d_i - 1) \cdot LB_i)$. However, this could be improved. The greatest integer β that satisfies $(\beta - \overline{t_i})^2 \le Z_i$ is calculated. Then, the β-constrained lower bound on the value of the objective function for product i, βTLB_i, is calculated: if $\beta TLB_i \le Z_i$ then $UB_i = \min(D - ((d_i - 1) \cdot LB_i); \beta)$, otherwise $\beta = \beta - 1$ and so on. Let us present a brief example: Let $D = 52$, $d_i = 5$, $\overline{t_i} = 52/5 = 10.4$ and $Z_i = 7.6$. Then, $\beta = 13$ and, when the distance between two units is fixed to $\beta = 13$, the best decomposition vector is $\lambda = (13,10,10,10,9)$, which gives $\beta TLB_i = 9.2 > Z_i$. Then, $\beta = 13 - 1 = 12$ and the best decomposition vector is $\lambda = (12,10,10,10,10)$, which gives $\beta TLB_i = 3.2 \le Z_i$ and $UB_i = \min(D - ((d_i - 1) \cdot LB_i); 12)$.

TD$_3$: E_{ik} and L_{ik} are tightened:

$$E_{ik} = \max(1 + LB_i \cdot (k-1); D - UB_i \cdot (d_i - k + 1) + 1) \qquad (i = 1,...,n; k = 1,...,d_i)$$
$$L_{ik} = \min(D - LB_i \cdot (d_i - k); UB_i \cdot k) \qquad (i = 1,...,n; k = 1,...,d_i)$$

TD$_4$: Variables sl_{ik} are defined, constraint set (7) is added and constraints (4) and (5) are replaced by constraints (8) and (9) respectively:

$$\sum_{h \in H_{ik}} h \cdot y_{ikh} = sl_{ik} \qquad (\forall i \in G1; k = 1,...,d_i) \qquad (7)$$
$$sl_{i,k+1} - sl_{i,k} = \delta_{ik}^{1} + ... + j \cdot \delta_{ik}^{j} + ... + UB_i \cdot \delta_{ik}^{UB_i} \qquad (\forall i \in G1; k = 1,...,d_i - 1) \qquad (8)$$
$$D - sl_{i,d_i} + sl_{i,1} = \delta_{i,d_i}^{1} + ... + j \cdot \delta_{i,d_i}^{j} + ... + UB_i \cdot \delta_{i,d_i}^{UB_i} \qquad (\forall i \in G1) \qquad (9)$$

TD$_5$ (to be used together with TD$_4$): Variables x_{ikh} are used and constraint sets (2), (3) and (7) are replaced by constraints (10), (11) and (12) respectively:

$$\sum_{\forall(i,k)\,|\,h=E_{ik}} x_{i,k,E_{ik}} + \sum_{\forall(i,k)\,|\,h\in H_{ik}\wedge h\neq E_{ik}\wedge h\neq L_{ik}} \left(x_{ik,h} - x_{ik,h-1}\right) + \sum_{\forall(i,k)\,|\,h=L_{ik}} \left(1 - x_{i,k,L_{ik}-1}\right) = 1 \qquad \left(h = 1,...,D\right) \quad (10)$$

$$x_{ik,h} \le x_{ik,h+1} \qquad \left(i = 1,...,n; k = 1,...,d_i; h = E_{ik},...,L_{ik}-2\right) \quad (11)$$

$$E_{ik}\cdot x_{i,k,E_{ik}} + \sum_{h=E_{ik}+1}^{L_{ik}-1} h\cdot\left(x_{ik,h} - x_{ik,h-1}\right) + L_{ik}\cdot\left(1 - x_{ik,L_{ik}-1}\right) = sl_{ik} \qquad \left(\forall i \in G1; k = 1,...,d_i\right) \quad (12)$$

TD$_6$: If the number of products with $d_i = 1$ is ≥ 2, a fictitious product, s, is defined with a number of units, $d_s = n - |G1|$. Variables y_{ikh} are defined only $\forall i \in G1$ and variables $y1_h$ $\left(h = 1,...,D\right)$ are defined. Constraints (2) and (3) are replaced by constraints (13) and (14) respectively and constraint set (15) is added:

$$\sum_{\forall(i,k)\,|\,i\in G1\wedge h\in H_{ik}} y_{ikh} + y1_h = 1 \qquad \left(h = 1,...,D\right) \quad (13)$$

$$\sum_{h\in H_{ik}} y_{ikh} = 1 \qquad \left(i \in G1; k = 1,...,d_i\right) \quad (14)$$

$$\sum_{h=1}^{D} y1_h = d_s \quad (15)$$

TD$_7$ (to be used together with TD$_1$ and TD$_2$): The values of LB_i and UB_i are calculated using $Z-2$ instead of Z (i.e., using $Z_i - 2$ instead of Z_i). If result of the CPLEX optimiser is "infeasible", $\bar{X}$ is optimal. This is because the difference between the values of the objective function corresponding to any pair of feasible solutions is an even integer number (Corominas *et al.*, 2004) and Z is the RTV value of $\bar{X}$.

3.2. Eliminating symmetries

ES$_1$: The first unit of one product i with the largest d_i is fixed in the first position of the sequence. We can assume, without loss of generality, that this product is product 1.

ES$_2$ (to be used together with TD$_4$): Two values are calculated for the sequence, considering it, respectively, in clockwise order $\left(S_1\right)$ and in anticlockwise order $\left(S_2\right)$. Then, it is imposed that the value for S_1 is less than or equal to the value for S_2. This value is calculated in two ways, which give the two following constraints:

$$\textbf{ES}_{2/1}\text{:}\ \sum_{i\in G1}\sum_{k=1}^{d_i} i\cdot sl_{ik} \le \sum_{i\in G1}\sum_{k=1}^{d_i} i\cdot\left(D - sl_{ik} + 1\right) \quad (16)$$

(i.e., the value is equal to the sum, for all positions occupied by the products $i \in G1$, of the result of multiplying the position number by the corresponding i).

ES2/2 (to be used together with ES1; i.e., product 1 occupies first position):

$$\sum_{k=1}^{d_1-1} k^2 \cdot \left(sl_{1,k+1} - sl_{1k}\right) + {d_1}^2 \cdot \left(D - sl_{1d_1} + sl_{11}\right) \leq \sum_{k=1}^{d_1-1} \left(d_1 - k + 1\right)^2 \cdot \left(sl_{1,k+1} - sl_{1k}\right) + 1^2 \cdot \left(D - sl_{1d_1} + sl_{11}\right) \quad (17)$$

(i.e., the value is equal to the sum, for all the units of product 1, of the result of multiplying the square of the ordinal number of a unit, k, by the distance between units k and $k+1$, considering d_1+1 equivalent to 1).

For example, let $D=10$, $d_1=3$ and the solution Sol_1 in which $sl_{11}=1$, $sl_{12}=6$ and $sl_{13}=8$ (with distances between units equal to 5, 2 and 3). The following symmetric solution can be obtained if the anticlockwise order is considered, Sol_2 in which $sl_{11}=1$, $sl_{12}=4$ and $sl_{13}=6$ (with distances between units equal to 3, 2 and 5). Constraint 17 allows eliminating the symmetric solution Sol_2, because the left-hand and right-hand values of constraint 17 are 40 and 56 for Sol_1 and 56 and 40 for Sol_2 (constraint 17 is not fulfilled by Sol_2).

ES3 (to be used together with TD4): A value is calculated for the sequence by shifting the units of product i with the largest d_i. Then, it is imposed that the value of the first sequence is less than or equal to the "value" of the other sequences. This value is calculated as the sum of the square of the number of the unit k, multiplied by the distance between units k and $k+1$ of product i:

$$\sum_{k=1}^{d_i-1} k^2 \cdot \left(sl_{i,k+1} - sl_{ik}\right) + {d_i}^2 \cdot \left(D - sl_{id_i} + sl_{i1}\right) \leq \sum_{k=1}^{d_i-1} \left(1 + \left(\left(k+j\right) \bmod d_i\right)\right)^2 \cdot \left(sl_{i,k+1} - sl_{ik}\right) + \left(1+j\right)^2 \cdot \left(D - sl_{id_i} + sl_{i1}\right) \quad \left(j = 0, \ldots, d_i - 2\right) \quad (18)$$

For example, let $D=10$, $d_1=3$ and the solution Sol_1 in which $sl_{11}=1$, $sl_{12}=6$ and $sl_{13}=8$ (with distances between units equal to 5, 2 and 3). The following two equivalent solutions can be obtained by shifting the units of product 1 (product with the largest d_i): Sol_2 in which $sl_{11}=1$, $sl_{12}=4$ and $sl_{13}=9$ (with distances between units equal to 3, 5 and 2); and Sol_3 in which $sl_{11}=1$, $sl_{12}=3$ and $sl_{13}=6$ (with distances between units equal to 2, 3, and 5). Constraint set 18 allows eliminating the equivalent solutions Sol_2 and Sol_3, because it is not fulfilled by these solutions.

3.3. Introducing redundant constraints

RC1 (to be used together with TD4): The domain of the variable sl_{ik} is constrained:

$$E_{ik} \le sl_{ik} \le L_{ik} \qquad (\forall i \in G1; k = 1,..., d_i) \qquad (19)$$

RC$_2$ (to be used together with TD$_4$): For each type of product $i \in G1$, it is imposed, over variables sl_{ik}, that the sum of the distances between its units is equal to D:

$$\sum_{k=1}^{d_i-1} \left(sl_{i,k+1} - sl_{i,k} \right) + \left(D - sl_{i,d_i} + sl_{i,1} \right) = D \qquad (\forall i \in G1) \qquad (20)$$

RC$_3$: For each unit k of product i $(i \in G1; k = 1,..., d_i - 1)$, it is imposed that the sum of variables $y_{i,k+1,h}$ until any position p of the sequence is less than or equal to the sum of variables y_{ikh} until position p. This idea arose from studying the solution to the linear program (LP), which can be obtained when the integrity of the variables is relaxed: y_{ikh} usually takes a fractional value and it is not unusual for $\sum_{h=E_{i,k+1}}^{p} y_{i,k+1,h}$ to take a value greater than $\sum_{h=E_{ik}}^{p} y_{ikh}$.

$$\sum_{h=E_{i,k+1}}^{p} y_{i,k+1,h} \le \sum_{h=E_{ik}}^{p} y_{ikh} \qquad (\forall i \in G1; k = 1,..., d_i - 1; p = E_{i,k+1},..., L_{i,k}) \qquad (21)$$

RC$_4$ (to be used together with OS$_1$): The consistency of values for γ_{ik}^{j} is imposed (although it is already guaranteed for the convexity of the objective function):

$$\gamma_{ik}^{j} \ge \gamma_{ik}^{j+1} \qquad (\forall i \in G1; k = 1,..., d_i; j = 1,..., UB_i - LB_i - 1) \qquad (22)$$

RC$_5$: A new consistency of values for y_{ikh} is imposed:

$$y_{ikh} + y_{i,k+1,j} \le 1 \qquad (\forall i \in G1; k = 1,..., d_i - 1; \forall (h, j) \in \overline{S}_{ik}) \qquad (23)$$

$$y_{id_ih} + y_{i,1,j} \le 1 \qquad (\forall i \in G1; \forall (h, j) \in \overline{S}_{id_i}) \qquad (24)$$

RC$_6$ (to be used together with OS$_1$; see Section 3.5): For each type of product $i \in G1$, it is imposed, over variables γ_{ik}^{j}, that the sum of the distances between its units is equal to D:

$$\sum_{j=1}^{UB_i-LB_i} \sum_{k=1}^{d_i} \gamma_{ik}^{j} = D - d_i \cdot LB_i \qquad (\forall i \in G1) \qquad (25)$$

$$\sum_{j=1}^{UB_i-LB_i} \sum_{k=1}^{d_i} \left(1 - \gamma_{ik}^{j}\right) = d_i \cdot UB_i - D \qquad (\forall i \in G1) \qquad (26)$$

3.4. Setting the parameters of CPLEX

SP$_1$: The "Absolute MIP gap tolerance" is set to $2-\varepsilon$. This is because the value of the objective function increases by multiples of 2, see Corominas *et al*. (2004).

SP$_2$: The "MIP Gomory fractional cuts indicator" is changed from the "Automatically determined" default value to "Generate Gomory fractional cuts aggressively".

3.5. Other strategies

OS$_1$: The well-known separable convex programming technique (see, e.g., Wagner, 1969) is applied. Variables δ_{ik}^{j} are replaced by variables γ_{ik}^{j}, objective function (1) is replaced by (27) and constraints (4), (5) and (6) are replaced by (28) and (29):

$$[MIN]\,RTV = \sum_{\forall i \in G1, k, j} \left(\left(LB_i + j \right)^2 - \left(LB_i + j - 1 \right)^2 \right) \cdot \gamma_{ik}^{j} + \sum_{i \in G1} d_i \cdot LB_i^{\,2} - \sum_{i \in G1} d_i \cdot \overline{t}_i^{\,2} \quad (27)$$

$$\sum_{h \in H_{i,k+1}} h \cdot y_{i,k+1,h} - \sum_{h \in H_{ik}} h \cdot y_{ikh} = LB_i + \gamma_{ik}^{1} + \ldots + \gamma_{ik}^{j} + \ldots + \gamma_{ik}^{UB_i - LB_i}$$

$$\left(\forall i \in G1; k = 1, \ldots, d_i - 1 \right) \quad (28)$$

$$D - \sum_{h \in H_{id_i}} h \cdot y_{i,d_i,h} + \sum_{h \in H_{i1}} h \cdot y_{i1h} = LB_i + \gamma_{i,d_i}^{1} + \ldots + \gamma_{i,d_i}^{j} + \ldots + \gamma_{i,d_i}^{UB_i - LB_i} \qquad \left(\forall i \in G1 \right) \quad (29)$$

If TD$_4$ is used, constraints (4), (5) and (6) are replaced by (30) and (31):

$$sl_{i,k+1} - sl_{i,k} = LB_i + \gamma_{ik}^{1} + \ldots + \gamma_{ik}^{j} + \ldots + \gamma_{ik}^{UB_i - LB_i} \qquad \left(\forall i \in G1; k = 1, \ldots, d_i - 1 \right) \quad (30)$$

$$D - sl_{i,d_i} + sl_{i,1} = LB_i + \gamma_{i,d_i}^{1} + \ldots + \gamma_{i,d_i}^{j} + \ldots + \gamma_{i,d_i}^{UB_i - LB_i} \qquad \left(\forall i \in G1 \right) \quad (31)$$

Variable γ_{ik}^{j} can be defined in two ways:

OS$_{1/1}$: γ_{ik}^{j} is defined as float $\left(0 \le \gamma_{ik}^{j} \le 1 \right)$.

OS$_{1/2}$: γ_{ik}^{j} is defined as binary $\left(\gamma_{ik}^{j} \in \{0,1\} \right)$.

OS$_2$ (to be used together with TD$_4$): The all-different operator of Constraint Logic Programming is included (all-different (sl_{ik})) and the ILOG Solver 6.0 is used to solve the mathematical problem. Variables $y_{ikh} \in \{0,1\}$ and constraint sets (2) and (3) are eliminated and constraint set (7) is replaced by constraint set (19).

4. Computational experiment

This section presents the computational experiments we carried out. First, we introduce the instances used. We then present a computational experiment that focuses on showing the improvements obtained using the ideas proposed in Section 3. Finally, we show a new computational experiment and the resulting MILP-based algorithm. We aim to show the effectiveness of the final MILP-based algorithm and obtain new optimal solutions in order to compare heuristic solutions.

4.1. Instances

Sixty instances were generated as follows. D was randomly selected with a uniform distribution between 20 and 30, between 30 and 35 and between 35 and 40, for instances 01 to 20, 21 to 40 and 41 to 60 respectively. For instances 01 to 20, n and d_i were randomly selected with a uniform distribution between 3 and $\left\lceil D/2 \right\rceil$ and between 1 and $\left\lceil (D-n+1)/2 \right\rceil$ (with $\sum_{i=1}^{n} d_i = D$) respectively. For instances 21 to 60, n and d_i were randomly selected with a uniform distribution between 3 and 12 and between 1 and $\left\lfloor \frac{D-n+1}{2.5} \right\rfloor$ (with $\sum_{i=1}^{n} d_i = D$) respectively.

4.2. Computational experiment 1: testing the proposed ideas

Just 20 instances were used to test the proposed ideas. A brief initial computational experiment was carried out and the instances whose computational time to guarantee the optimal solution was neither very short nor very long were selected. Table 1 shows the code of the instance (CI) and the values of D, n and d_i.

NI	*D*	*n*	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
1	28	11	7	5	4	3	3	1	1	1	1	1	1	
5	24	3	11	8	5									
9	27	6	9	8	5	3	1	1						
11	21	5	7	7	3	2	2							
21	34	11	7	7	6	4	3	2	1	1	1	1	1	
23	32	4	15	7	5	5								
27	34	12	5	5	5	5	3	3	2	2	1	1	1	1
29	35	8	10	6	5	5	3	3	2	1				
31	30	9	6	5	5	4	4	3	1	1	1			
33	31	7	10	5	5	4	4	2	1					
34	30	3	13	10	7									
37	35	10	9	6	5	5	3	2	2	1	1	1		
39	32	11	7	6	4	4	3	2	2	1	1	1	1	
40	30	12	5	4	4	4	3	3	2	1	1	1	1	1
41	35	10	9	9	5	3	3	2	1	1	1	1		
44	38	7	9	9	8	5	5	1	1					
45	40	10	11	10	9	3	2	1	1	1	1	1		
50	38	8	11	10	6	5	3	1	1	1				
52	37	10	9	8	8	5	2	1	1	1	1	1		
57	37	3	15	12	10									

Table 1. Data of the 20 instances

The MILP models were tested to optimality using the ILOG CPLEX 9.0 optimiser and a Pentium Centrino at 1.6 GHz with 512 Mb of RAM. The maximum calculation time for each instance was set at 2,000 seconds. The strategy for testing the proposed ideas was to incorporate or discard each one depending on its results, in an accumulative manner.

Table 2 shows the results of ideas that improved the calculation time. The first three columns are code of the instance (CI), value of the optimal solution (Z^*) and

calculation time for MILP-0 $(M0)$ respectively (if an instance was not solved in 2,000 seconds, the calculation time is listed as 2,000 seconds). For each idea, the calculation time is shown in the other columns. The bottom four rows show the sum of the calculation times (ΣCT), the number of instances that proved optimal (NZ^*), and the percentage decrease of the calculation time using the idea presented $(\%\nabla)$ and using model MILP-0 $(\Sigma\%\nabla)$ respectively.

CI	Z^*	$M0$	TD_3 + SP_1	ES_1	TD_1 + TD_2	TD_7	TD_4	$OS_{1/2}$	$ES_{2/2}$	ES_3	RC_4	TD_6	RC_6
1	6.53	2000	2000	520	326	209	128	13	12	7	6	2	3
5	10.44	2000	2000	854	458	220	36	10	3	2	1	1	2
9	11.08	2000	2000	2000	2000	2000	2000	151	200	86	65	41	30
11	9	2000	2000	795	321	387	69	36	20	3	3	3	2
21	6.71	2000	2000	2000	1465	2000	330	32	11	5	8	4	4
23	11.85	2000	2000	2000	2000	2000	621	274	46	12	8	8	6
27	4.53	603	602	159	10	5	2	1	1	1	2	6	1
29	9.17	2000	2000	2000	2000	2000	2000	949	440	306	213	213	187
31	6	2000	2000	2000	1381	1383	100	65	34	11	19	24	23
33	6.50	2000	2000	1369	17	3	2	2	1	0	0	0	0
34	10.20	2000	2000	2000	788	135	24	10	5	1	1	1	1
37	7.39	2000	2000	2000	1125	888	262	170	100	74	19	77	23
39	5.71	2000	2000	2000	360	158	78	596	20	19	8	11	9
40	5	2000	2000	2000	89	88	51	191	116	23	72	6	13
41	7.61	2000	2000	2000	1524	1377	232	62	14	30	28	12	9
44	11.11	2000	2000	2000	2000	2000	2000	791	164	42	113	145	53
45	9.43	2000	2000	2000	2000	2000	2000	279	246	29	34	19	15
50	9.53	2000	2000	2000	1793	34	117	50	21	0	0	0	0
52	10.34	2000	2000	2000	2000	2000	419	78	175	41	66	18	15
57	10.75	2000	2000	2000	505	504	140	9	7	2	2	2	2
	ΣCT	38603	38602	33697	22162	19391	10611	3769	1636	694	668	593	398
	NZ^*	1	1	5	14	13	16	20	20	20	20	20	20
	$\%\nabla$		0.00	12.71	34.23	12.50	45.28	64.48	56.59	57.58	3.75	11.23	32.88
	$\Sigma\%\nabla$		0.00	12.71	42.59	49.77	72.51	90.24	95.76	98.20	98.27	98.46	98.97

Table 2. Results corresponding to the ideas for improving the calculation time

The results show a huge improvement gained by the best mathematical program, the one includes the TD_3, SP_1, ES_1, TD_1, TD_2, TD_7, TD_4, $OS_{1/2}$, $ES_{2/2}$, ES_3, RC_4, TD_6 and RC_6. The total calculation time dropped from 38,603 seconds (this time would have been even longer if the calculation time of each instance had not been limited to 2,000 seconds) to just 398 seconds. Moreover, all 20 instances were solved to optimality. When the optimal solution is not guaranteed in 2,000 seconds, the feasible solution and the lower bound are usually of equal or better quality than in the previous model. Finally, idea RC_2 did not significantly change the total calculation time and the number of constraints increased. The importance of modelling was demonstrated, as well as the huge impact that redundant constraints and the elimination of symmetries have on the effectiveness of MILPs.

The results of the computational experiment show that the other ideas tested (TD_5, $ES_{2/1}$, RC_1, RC_3, RC_5, SP_2, $OS_{1/1}$ and OS_2) performed worse: the sum of the calculation times increased and the number of instances that proved optimal decreased.

4.3. Computational experiment 2: solving the 60 instances

The 60 instances generated were tested using the MILP-based algorithm with a maximum calculation time of 2,000 seconds. This computational experiment aimed to show the effectiveness of the final MILP-based algorithm and to obtain new optimal solutions in order to compare heuristic solutions.

The following is the MILP-based algorithm:

1. Obtain $\overline{X}$ (initial feasible solution) and Z (RTV value of $\overline{X}$).
2. Define a fictitious product s with a number of units, d_s, equal to n minus the cardinal of $G1$.
3. $Z = Z - 2$, according to TD_7 (if the CPLEX result is "infeasible", $\overline{X}$ is optimal). Calculate LB_i, UB_i, E_{ik} and L_{ik} according to TD_1, TD_2 and TD_3.
4. Fix the first unit of product i with the largest d_i in the first position of the sequence (according to ES_1).
5. Set the "Absolute MIP gap tolerance" of CPLEX to 1.99999 (according to SP_1). Set the maximum calculation time of CPLEX to 2,000 seconds.
6. Solve the MILP using CPLEX.

The following is the MILP to be tested to optimality:

Variables:

sl_{ik}	Position of unit k of product i $(\forall i \in G1; k = 1,...,d_i)$
$y_{ikh} \in \{0,1\}$	1 if and only if unit k of product i is placed in position h $(\forall i \in G1$ $k = 1,...,d_i;\ h \in H_{ik})$
$y1_h \in \{0,1\}$	1 if and only if a unit of any product with $d_i = 1$ is placed in position h $(h = 1,...,D)$
$\gamma_{ik}^j \in \{0,1\}$	1 if and only if the distance between units k and $k+1$ of product i is greater than or equal to $LB_i + j$ $(\forall i \in G1; k = 1,...,d_i; j = 1,...,UB_i - LB_i)$

Model:

$$[MIN]\, RTV = \sum_{\forall i \in G1, k, j} \left(\left(LB_i + j \right)^2 - \left(LB_i + j - 1 \right)^2 \right) \cdot \gamma_{ik}^j + \sum_{i \in G1} d_i \cdot LB_i^2 - \sum_{i \in G1} d_i \cdot \overline{t}_i^2 \qquad (27)$$

$$\sum_{h \in H_{ik}} h \cdot y_{ikh} = sl_{ik} \qquad (\forall i \in G1; k = 1,...,d_i) \qquad (7)$$

$$sl_{i,k+1} - sl_{i,k} = LB_i + \gamma_{ik}^1 + ... + \gamma_{ik}^j + ... + \gamma_{ik}^{UB_i - LB_i} \qquad (\forall i \in G1; k = 1,...,d_i - 1) \qquad (30)$$

$$D - sl_{i,d_i} + sl_{i,1} = LB_i + \gamma_{i,d_i}^1 + ... + \gamma_{i,d_i}^j + ... + \gamma_{i,d_i}^{UB_i - LB_i} \qquad (\forall i \in G1) \qquad (31)$$

$$\gamma_{ik}^j \geq \gamma_{ik}^{j+1} \qquad (\forall i \in G1; k = 1,...,d_i; j = 1,...,UB_i - LB_i - 1) \qquad (22)$$

$$\sum_{\forall (i,k) | i \in G1 \wedge h \in H_{ik}} y_{ikh} + y1_h = 1 \qquad (h = 1,...,D) \qquad (13)$$

$$\sum_{h \in H_{ik}} y_{ikh} = 1 \qquad (i \in G1; k = 1,...,d_i) \qquad (14)$$

$$\sum_{h=1}^{D} y1_h = d_s \tag{15}$$

$$\sum_{k=1}^{d_i-1} k^2 \cdot \left(sl_{i,k+1} - sl_{ik}\right) + d_i{}^2 \cdot \left(D - sl_{id_i} + sl_{i1}\right) \leq \sum_{k=1}^{d_i-1} \left(d_i - k + 1\right)^2 \cdot \left(sl_{i,k+1} - sl_{ik}\right) + 1^2 \cdot \left(D - sl_{id_i} + sl_{i1}\right) \tag{17}$$

$$\sum_{k=1}^{d_i-1} k^2 \cdot \left(sl_{i,k+1} - sl_{ik}\right) + d_i{}^2 \cdot \left(D - sl_{id_i} + sl_{i1}\right) \leq \sum_{k=1}^{d_i-1} \left(1 + \left(\left(k + j\right) \bmod d_i\right)\right)^2 \cdot \left(sl_{i,k+1} - sl_{ik}\right) + \left(1 + j\right)^2 \cdot \left(D - sl_{id_i} + sl_{i1}\right) \quad \left(j = 0, \ldots, d_i - 2\right) \tag{18}$$

$$\sum_{j=1}^{UB_i - LB_i} \sum_{k=1}^{d_i} \gamma_{ik}^{j} = D - d_i \cdot LB_i \qquad \left(\forall i \in G1\right) \tag{25}$$

$$\sum_{j=1}^{UB_i - LB_i} \sum_{k=1}^{d_i} \left(1 - \gamma_{ik}^{j}\right) = d_i \cdot UB_i - D \qquad \left(\forall i \in G1\right) \tag{26}$$

Variables sl_{ik} and constraint set (7) are defined according to TD_4. Variables y_{ikh} are redefined, variables $y1_h$ are defined and constraints (13), (14) and (15) are used according to TD_6. Variables γ_{ik}^{j}, objective function (27) and constraints (30) and (31) are introduced according to $OS_{1/2}$. Constraint set (22) is necessary according to RC_4. Constraints (17) and (18) are included according to $ES_{2/2}$ and ES_3, respectively. And constraints (25) and (26) are defined according to RC_6.

Table 3 shows the results. For each instance, the following information is shown: code of the instance (CI), calculation time $(Time)$, the guaranteed optimal value of the objective function solution (in column *Sol* when column *Bnd* is empty) or the best solution obtained (Sol) and the best bound obtained (Bnd) if the maximum calculation time (2,000 seconds) is reached.

NI	*Time*	*Sol*	*Bnd*
1	3.13	6.53	
2	0.16	3	
3	0.00	6.18	
4	0.17	4.53	
5	1.52	10.44	
6	0.23	4.05	
7	0.23	4.86	
8	0.12	4.05	
9	30.10	11.08	
10	0.01	1.33	
11	1.84	9	
12	0.53	9.07	
13	0.02	1.25	
14	0.44	6.41	
15	6.94	12.00	
16	0.26	6.67	
17	0.02	3.73	
18	0.58	6.23	
19	0.17	4.75	

NI	*Time*	*Sol*	*Bnd*
21	3.70	6.71	
22	1,868	10.67	
23	5.89	11.85	
24	99.17	10.89	
25	33.28	13.64	
26	3.73	6.47	
27	1.50	4.53	
28	0.00	7.80	
29	187.36	9.17	
30	469.72	14.70	
31	22.89	6	
32	0.05	4.88	
33	0.11	6.50	
34	1.13	10.20	
35	1,719	15.69	
36	19.89	12.91	
37	23.11	7.39	
38	33.98	12.89	
39	9.08	5.71	

NI	*Time*	*Sol*	*Bnd*
41	8.89	7.61	
42	365.53	10.80	
43	2,000	14.89	11.69
44	53.45	11.11	
45	14.94	9.43	
46	2,000	20.28	16.28
47	2,000	27.48	9.34
48	167.44	13.98	
49	2,000	17.58	9.58
50	0.42	9.53	
51	15.92	14.25	
52	15.06	10.34	
53	2,000	24.22	8.22
54	2,000	20.49	10.49
55	57.20	6.80	
56	104.95	11.27	
57	2.31	10.75	
58	2,000	23.55	11.84
59	85.92	14.61	

20	0.05	4.28		**40**	12.95	5		**60**	817.03	9.41	

Table 3. Results for 60 instances

The results show that the practical limit for obtaining optimal solutions increased from $D = 25$ to around $D = 40$ units to be scheduled.

The final computational experiment was carried out using the instances that cannot be solved optimally in 2,000 seconds. The calculation time increased to 20,000 seconds. Table 4 shows the results. For instances 43, 46 and 58, the optimal solution can be guaranteed. In the other instances, the solution and bound improved when the calculation time increased tenfold.

NI	*Time*	*Sol*	*Bnd*
43	3,278	14.89	
46	3,209	20.28	
47	20,000	25.48	11.48
49	20,000	17.58	13.58
53	20,000	22.22	10.22
54	20,000	18.49	12.49
58	15,595	23.55	

Table 4. Results for unsolved instances

5. Conclusions

This paper deals with the optimal solution of the Response Time Variability Problem (RTVP) by means of mathematical programming. The RTVP is a new scheduling problem with a broad range of real-life applications that is very difficult to solve to optimality. Corominas *et al*. (2004) have tested some mathematical programming models (MILPs) to solve the RTVP to optimality, but the practical limit for obtaining optimal solutions has been 25 units with these models.

We analysed the special features of the RTVP and proposed and tested some ideas for improving the best mathematical programming models presented in the literature. The size of the instances that can be solved to optimality increased from 25 to around 40. We obtained new optimal solutions to the RTVP that can be used to compare the results of heuristic procedures. Finally, we showed the importance of modelling and the impact that reformulation, redundant constraints and the elimination of symmetries have on the effectiveness of MILPs.

Future research may focus on designing branch and bound, and other that those presented by Corominas *et al*. (2004) heuristics and metaheuristics for the RTV problem.

References

[1] Anily S, Glass CA, Hassin R. The scheduling of maintenance service. *Discrete Applied Mathematics* 82, 27-42. 1998.

[2] Bar-Noy A, Bhatia R, Naor J, Schieber B. Minimizing service and operation costs of periodic scheduling. *Mathematics of Operations Research* 27, 518-544. 2002.

[3] Corominas A, Kubiak W, Moreno N. Response time variability. *Working paper IOC-DT-P-2004-08*. UPC, Barcelona, 2004. (to be published in Journal of Scheduling)
[4] Dong L, Melhem R, Mosse D. Time slot allocation for real-time messages with negotiable distance constrains requirements. The Real-time Technology and Application Symposium, RTAS, Denver, CO. 1998.
[5] Han CC, Lin KJ, Hou CJ. Distance-constrained scheduling and its applications in real-time systems. *IEEE Trans. on Computers* 45 (7), 814-826. 1996.
[6] León D, Corominas A, Lusa A. Resolución del problema PRV min-var. *Working paper IOC-DT-I-2003-03*. UPC, Barcelona, 2003.
[7] Monden Y. *Toyota Production Systems*. Industrial Engineering and Management Press, Norcross, GA, 1983.
[8] Wagner HM. *Principles of operations research-with applications to managerial decisions*. Prentice Hall, Englewood Cliffs, NJ, 1969.
[9] Waldspurger CA, Weihl WE. Stride scheduling: deterministic proportional-share resource management. *Technical Memorandum MIT/LCS/TM-528*. MIT, Laboratory for Computer Science, Cambridge, 1995.